## Claims

What is claimed is:

1. A processor, comprising M independent vector register files, said M vector register files adapted to collectively store a matrix of L data elements, each data element having B binary bits, said matrix having N rows and M columns, said L=N*M, each column having K subcolumns, said $N \geq 2$, said $M \geq 2$, said $K \geq 1$, said $B \geq 1$, each row of said N rows being addressable, each subcolumn of said K subcolumns being addressable, said processor not adapted to duplicatively store said L data elements.

2. The processor of claim 1, wherein the processor further comprises M address registers, wherein each address register of the M address registers is associated with a corresponding one of the M vector register files, wherein each of the M vector register files includes an array of N registers, wherein each of the N*M registers of the M vector register files are adapted to store a data element of the L data elements, and wherein each vector register file is independently addressable through its associated address register being adapted to point to one of the N registers of said vector register file.

3. The processor of claim 2, wherein the data elements of each subcolumn are adapted to be stored in different vector register files, and wherein the data elements of each row are adapted to

be stored in different vector register files.

4. The processor of claim 3, wherein the data elements of each subcolumn are adapted to be stored in different relative register locations of the different vector register files, and wherein the data elements of each row are adapted to be stored in a same relative register location of the different vector register files.

5. The processor of claim 3, wherein the processor further comprises M multiplexors respectively coupled to the M vector register files, and wherein if the matrix is stored in the M vector register files then:

the M multiplexors are adapted to respond to a command to read a row of the matrix by mapping the data elements of the row from the M vector register files to the row of the matrix in accordance with a read-row mapping algorithm; and

the M multiplexors are adapted to respond to a command to read a subcolumn of the matrix by reading the data elements of the subcolumn from the M vector register files to the subcolumn of the matrix in accordance with a read-subcolumn mapping algorithm.

6. The processor of claim 3,

wherein the processor further comprises M multiplexors respectively coupled to the M vector register files;

wherein the M multiplexors are adapted to respond to a command to write a row of the

matrix by mapping the data elements of the row to the M vector register files in accordance with a write-row mapping algorithm; and

wherein the M multiplexors are adapted to respond to a command to write a subcolumn of the matrix by mapping the data elements of the subcolumn to the M vector register files in accordance with a write-subcolumn mapping algorithm.

7. The processor of claim 2, wherein the processor further comprises M multiplexors respectively coupled to the M vector register files such that each of the M multiplexors has a different value.

8. The processor of claim 1, wherein the matrix of L data elements are stored in the M vector register files.

9. A method for processing matrix data, comprising:

providing the processor; and

providing M independent vector register files within the processor, said M vector register files collectively storing a matrix of L data elements, each data element having B binary bits, said matrix having N rows and M columns, said L=N*M, each column having K subcolumns, said $N \geq 2$, said $M \geq 2$, said $K \geq 1$, said $B \geq 1$, each row of said N rows being addressable, each subcolumn of said K subcolumns being addressable, said processor not duplicatively storing said L data elements.

10. The method of claim 9, wherein the method further comprises providing M address registers within the processor, wherein each address register of the M address registers is associated with a corresponding one of the M vector register files, wherein each of the M vector register files includes an array of N registers, wherein each of the N*M registers of the M vector register files stores a data element of the L data elements, and wherein each vector register file is independently addressable through its associated address register being adapted to point to one of the N registers of said vector register file.

11. The method of claim 10, wherein the data elements of each subcolumn are stored in different vector register files, and wherein the data elements of each row are stored in different vector register files.

12. The method of claim 11, wherein the data elements of each subcolumn are stored in different relative register locations of the different vector register files, and wherein the data elements of each row are stored in a same relative register location of the different vector register files.

13. The method of claim 11, wherein the method further comprises providing M multiplexors respectively coupled to the M vector register files, and wherein if the matrix is stored in the M vector register files then:

the M multiplexors are adapted to respond to a command to read a row of the matrix by mapping the data elements of the row from the M vector register files to the row of the matrix in accordance with a read-row mapping algorithm; and

the M multiplexors are adapted to respond to a command to read a subcolumn of the matrix by reading the data elements of the subcolumn from the M vector register files to the subcolumn of the matrix in accordance with a read-subcolumn mapping algorithm.

14. The method of claim 11,

wherein the method further comprises providing M multiplexors respectively coupled to the M vector register files;

wherein the M multiplexors are adapted to respond to a command to write a row of the matrix by mapping the data elements of the row to the M vector register files in accordance with a write-row mapping algorithm; and

wherein the M multiplexors are adapted to respond to a command to write a subcolumn

of the matrix by mapping the data elements of the subcolumn to the M vector register files in accordance with a write-subcolumn mapping algorithm.

15. The method of claim 10, wherein the method further comprises providing M multiplexors respectively coupled to the M vector register files such that each of the M multiplexors has a different value.

16. The method of claim 9, further comprising addressing a row of the N rows.

17. The method of claim 9, further comprising addressing a subcolumn of the K*M subcolumns.

18. A processor, comprising M independent vector register files, said M vector register files adapted to collectively store a matrix of L data elements, each data element having B binary bits, said matrix having N rows and M columns, said L=N*M, each column having K subcolumns, said $N \geq 2$, said $M \geq 2$, said $K \geq 1$, said $B \geq 1$, each row of said N rows being addressable, each subcolumn of said K subcolumns being addressable, said matrix including a set of arrays such that each array is a row or subcolumn of the matrix, said processor adapted to execute an instruction that performs an operation on a first array of the set of arrays, said operation being performed with selectivity with respect to the data elements of the first array.

19. The processor of claim 18, wherein the processor further comprises M multiplexors respectively coupled to the M vector register files, and wherein the values associated with the M multiplexors control said selectivity.

20. The processor of claim 18, wherein the processor further comprises M address registers, wherein each address register of the M address registers is associated with a corresponding one of the M vector register files, wherein each of the M vector register files includes an array of N registers, wherein each of the N*M registers of the M vector register files are adapted to store a data element of the L data elements, and wherein each vector register file is independently addressable through its associated address register being adapted to point to one of the N registers of said vector register file.

21. The processor of claim 18, wherein the instruction is adapted to copy at least one data element of the first array of the set of arrays to a second array of the set of arrays, and wherein the instruction does not insert an exact copy of the first array into the second array.

22. The processor of claim 18, wherein the instruction is adapted to rearrange the data elements of the first array within the first array.

23. The processor of claim 18, wherein the processor is not adapted to duplicatively store the L data elements.

24. The processor of claim 18, wherein the matrix of L data elements are stored in the M vector register files.

25. A method for processing matrix data, comprising:

providing the processor;

providing M independent vector register files within the processor, said M vector register files collectively storing a matrix of L data elements, each data element having B binary bits, said matrix having N rows and M columns, said L=N*M, each column having K subcolumns, said $N \geq 2$, said $M \geq 2$, said $K \geq 1$, said $B \geq 1$, each row of said N rows being addressable, each subcolumn of said K subcolumns being addressable, said matrix including a set of arrays such that each array is a row or subcolumn of the matrix; and

executing an instruction by said processor, said instruction performing an operation on a first array of the set of arrays, said operation being performed with selectivity with respect to the data elements of the first array.

26. The method of claim 25, further comprising providing M multiplexors respectively coupled to the M vector register files, wherein the values associated with the M multiplexors control said selectivity.

27. The method of claim 25, wherein the method further comprises providing M address registers within the processor, wherein each address register of the M address registers is associated with a corresponding one of the M vector register files, wherein each of the M vector register files includes an array of N registers, wherein each of the N*M registers of the M vector register files stores a data element of the L data elements, and wherein each vector register file is

independently addressable through its associated address register being adapted to point to one of the N registers of said vector register file.

28. The method of claim 25, wherein said performing an operation includes copying at least one data element of the first array of the set of arrays to a second array of the set of arrays, and wherein said copying does not insert an exact copy of the first array into the second array.

29. The method of claim 25, wherein said performing an operation includes rearranging the data elements of the first array within the first array.

30. The method of claim 25, wherein the processor is not duplicatively storing the L data elements.